Internet Databases

Chapter 22
HTML

- Simple markup language
- Text is annotated with language commands called tags, usually consisting of a start tag and an end tag
HTML Example: Book Listing

<HTML><BODY>
Fiction:
<UL><LI>Author: Milan Kundera</LI>
    <LI>Title: Identity</LI>
    <LI>Published: 1998</LI>
</UL>

Science:
<UL><LI>Author: Richard Feynman</LI>
    <LI>Title: The Character of Physical Law</LI>
    <LI>Hardcover</LI>
</UL></BODY></HTML>
Web Pages with Database Contents

- Web pages contain the results of database queries. How do we generate such pages?
  - Web server creates a new process for a program interacts with the database.
  - Web server communicates with this program via CGI (Common gateway interface)
  - Program generates result page with content from the database
  - Other protocols: ISAPI (Microsoft Internet Server API), NSAPI (Netscape Server API)
Application Servers

- In CGI, each page request results in the creation of a new process: very inefficient
- Application server: Piece of software between the web server and the applications
- Functionality:
  - Hold a set of pre-forked threads or processes for performance
  - Database connection pooling (reuse a set of existing connections)
  - Integration of heterogeneous data sources
  - Transaction management involving several data sources
  - Session management
Other Server-Side Processing

- Java Servlets: Java programs that run on the server and interact with the server through a well-defined API.
- JavaBeans: Reusable software components written in Java.
- Java Server Pages and Active Server Pages: Code inside a web page that is interpreted by the web server.
Beyond HTML: XML

- Extensible Markup Language (XML): “Extensible HTML”
- Confluence of SGML and HTML: The power of SGML with the simplicity of HTML
- Allows definition of new markup languages, called document type declarations (DTDs)
XML: Language Constructs

- **Elements**
  - Main structural building blocks of XML
  - Start and end tag
  - Must be properly nested
- Element can have attributes that provide additional information about the element
- **Entities**: like macros, represent common text.
- **Comments**
- **Document type declarations (DTDs)**
Booklist Example in XML

```xml
<?xml version="1.0" standalone="yes"?>
<!DOCTYPE BOOKLIST SYSTEM "booklist.dtd">
<BOOKLIST>
  <BOOK genre="Fiction">
    <AUTHOR>
      <FIRST>Milan</FIRST><LAST>Kundera</LAST>
    </AUTHOR>
    <TITLE>Identity</TITLE>
    <PUBLISHED>1998</PUBLISHED>
  </BOOK>
  <BOOK genre="Science" format="Hardcover">
    <AUTHOR>
      <FIRST>Richard</FIRST><LAST>Feynman</LAST>
    </AUTHOR>
    <TITLE>The Character of Physical Law</TITLE>
  </BOOK>
</BOOKLIST>
```
XML: DTDs

- A DTD is a set of rules that defines the elements, attributes, and entities that are allowed in the document.
- An XML document is well-formed if it does not have an associated DTD but it is properly nested.
- An XML document is valid if it has a DTD and the document follows the rules in the DTD.
An Example DTD

<!DOCTYPE BOOKLIST [
  <!ELEMENT BOOKLIST (BOOK)>*
  <!ELEMENT BOOK (AUTHOR, TITLE, PUBLISHED?)>*
  <!ELEMENT AUTHOR (FIRST, LAST)>*
    <!ELEMENT FIRST (#PCDATA)>*
    <!ELEMENT LAST (#PCDATA)>*
  <!ELEMENT TITLE (#PCDATA)>*
  <!ELEMENT PUBLISHED (#PCDATA)>*
  <!ATTLIST BOOK genre (Science | Fiction) #REQUIRED>
    <!ATTLIST BOOK format (Paperback | Hardcover) “Paperback”>
]>

Domain-Specific DTDs

- Development of standardized DTDs for specialized domains enables data exchange between heterogeneous sources

- Example: Mathematical Markup Language (MathML)
  - Encodes mathematical material on the web
  - In HTML: `<IMG SRC="xysq.gif" ALT="(x+y)^2">`
  - In MathML:
    `<apply> <power/>`<br>    `<apply> <plus/> <ci>x</ci> <ci>y</ci> </apply>`<br>    `<cn>2</cn>`
    `</apply>`
XML-QL: Querying XML Data

- Goal: High-level, declarative language that allows manipulation of XML documents
- No standard yet
- Example query in XML-QL:
  WHERE
  <BOOK>
    <NAME><LAST>$1</LAST></NAME>
  </BOOK> in "www.booklist.com/books.xml
CONSTRUCT <RESULT> $1 </RESULT>
XML-QL (Contd.)

A more complicated example:

```
WHERE <BOOK> $b <BOOK> IN
    "www.booklist.com/books.xml",
    <AUTHOR> $n </AUTHOR>
    <PUBLISHED> $p </PUBLISHED> in $e
CONSTRUCT
    <RESULT>
        <PUBLISHED> $p </PUBLISHED>
        WHERE <LAST> $l </LAST> IN $n
        CONSTRUCT <LAST> $l </LAST>
    </RESULT>
```

Semi-structured Data

- Data with partial structure
- All data models for semi-structured data use some type of labeled graph
- We introduce the object exchange model (OEM):
  - Object is triple (label, type, value)
  - Complex objects are decomposed hierarchically into smaller objects
Example: Booklist Data in OEM

BOOK

AUTHOR

TITLE

PUBLISHED

AUTHOR

FORMAT

identity

1998

The character of physical law

Hardcover

Milan Kundera

Richard Feynman

Database Management Systems, 2nd Edition. R. Ramakrishnan and Johannes Gehrke

16
Indexing for Text Search

- Text database: Collection of text documents
- Important class of queries: Keyword searches
  - Boolean queries: Query terms connected with AND, OR and NOT. Result is list of documents that satisfy the boolean expression.
  - Ranked queries: Result is list of documents ranked by their “relevance”.
  - IR: Precision (percentage of retrieved documents that are relevant) and recall (percentage of relevant objects that are retrieved)
Inverted Files

- For each possible query term, store an ordered list (the inverted list) of document identifiers that contain the term.
- Query evaluation: Intersection or Union of inverted lists.
- Example: Agent AND James

<table>
<thead>
<tr>
<th>RID</th>
<th>Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Agent James</td>
</tr>
<tr>
<td>2</td>
<td>Mobile agent</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Word</th>
<th>Inverted List</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent</td>
<td>&lt;1,2&gt;</td>
</tr>
<tr>
<td>James</td>
<td>&lt;1&gt;</td>
</tr>
<tr>
<td>Mobile</td>
<td>&lt;2&gt;</td>
</tr>
</tbody>
</table>
Signature Files

- Index structure (the signature file) with one data entry for each document
- Hash function hashes words to bit-vector.
- Data entry for a document (the signature of the document) is the OR of all hashed words.
- Signature $S_1$ matches signature $S_2$ if $S_2 \& S_1 = S_2$
Signature Files: Query Evaluation

- **Boolean query consisting of conjunction of words:**
  - Generate query signature $S_q$
  - Scan signatures of all documents.
  - If signature $S$ matches $S_q$, then retrieve document and check for false positives.

- **Boolean query consisting of disjunction of $k$ words:**
  - Generate $k$ query signatures $S_1, \ldots, S_k$
  - Scan signature file to find documents whose signature matches any of $S_1, \ldots, S_k$
  - Check for false positives
# Signature Files: Example

<table>
<thead>
<tr>
<th>Word</th>
<th>Hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent</td>
<td>1010</td>
</tr>
<tr>
<td>James</td>
<td>1100</td>
</tr>
<tr>
<td>Mobile</td>
<td>0001</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RID</th>
<th>Document</th>
<th>Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Agent James</td>
<td>1110</td>
</tr>
<tr>
<td>2</td>
<td>Mobile agent</td>
<td>1011</td>
</tr>
</tbody>
</table>
Summary

- Publishing databases on the web requires server-side processing such as CGI-scripts, Servlets, ASP, or JSP.
- XML is an emerging document description standard that allows the definition of new DTDs. Query languages for XML documents such as XQL are emerging.
- Text databases have gained importance with the proliferation of text data on the web. Boolean queries can be efficiently evaluated using an inverted index or a signature file. Evaluation of ranked queries is a more difficult problem.